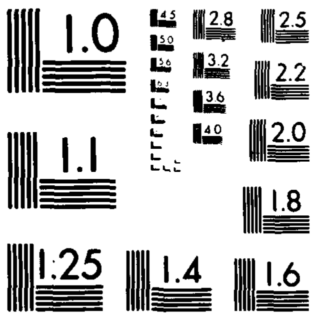
AD-A112 866   ADMIRALTY MARINE TECHNOLOGY ESTABLISHMENT TEDDINGTON--ETC F/G 5/7
FOUNDATIONS FOR THE DEVELOPMENT OF A SIMPLE NATURAL LANGUAGE IN--ETC(U)
JAN 82 M B ROWLEY
UNCLASSIFIED   AMTE(E)TM8110-2   DRIC-BR-82261   NL

1 of 1
AD A
112866

END
DATE
FILMED
4-82
DTIC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

AMTE(E) TM 81102

FOUNDATIONS FOR THE DEVELOPMENT OF A SIMPLE NATURAL LANGUAGE INTERFACE FOR TASK KNOWLEDGE ELICITATION AND REPRESENTATION

BY

M B ROWLEY

Summary

This report presents a state-of-the-art synopsis of the field of natural language understanding, including the relative importance of "meaning", "syntax" and "semantics". A definition of "parsing" is suggested and several existing parsers are discussed. The final section discusses the contribution of the various parsers and parsing techniques to the implementation of a natural language interface to be used in research on Intelligent Support System~ at APU.

AMTE (Teddington)
Queen's Road
TEDDINGTON Middlesex TW11 OLN

PAGES 54
FIGS. 16

January 1982

CONTENTS

INTRODUCTION

For a considerable number of years, the study of natural language has been the province of linguists and philosophers. More recently, however, researchers in the field of Artificial Intelligence (AI) have emphasised the importance of natural language to their discipline.

In the AI field, natural language research has taken two main directions, characterised by those wishing to simulate human cognitive processes in an attempt to arrive at a better understanding of human linguistic processing and those aiming to demonstrate machine intelligence, making no claims about the "psychological validity" of their model of natural language understanding.

The "first generation" of computer-based natural language systems, as Winograd (1974) points out, focussed on producing the appearance of language understanding in a highly constrained domain. There are several programs that illustrate this approach, the best known being ELIZA - a simulated Rogerian psychotherapist (Weizenbaum 1966), and PARRY - a simulated paranoid patient (Colby 1968). These programs are based on a fairly simple "key-word" matching process, that is, they search for certain words in the user's input which, if found, trigger standard responses. Programs such as these have enjoyed some success, but they appear to understand an extremely limited number of inputs, demonstrated simply by inputting sentences devoid of any key-words.

The "second generation" of computer-based natural language systems saw a shift in emphasis from the mere appearance of language understanding to actually modelling the basic mechanisms of language processing. The best known illustrations of this approach are SHRDLU (Winograd 1971), a program that manipulates objects in a simulated world of blocks and LUNAR (Woods 1974) which answers questions about samples brought back from the moon. Once again, the domains of such programs are very restricted. The question-answering technique, also based on key-word search, gained popularity because no complicated language generating component is required.

AMTE/APU's current approach aims primarily to demonstrate language understanding, with no emphasis on modelling the cognitive processes that may be involved. The underlying philosophy of this approach to the study of natural language understanding in AI is the idea that communicating with a computer program in one's own language is a particularly powerful demonstration of machine intelligence. In addition, considerable interest in "Intelligent Support Systems" is currently being shown by various research communities, Government Departments and certain industries and this has further emphasised the need for work on natural language systems that will aid human-machine communication, (see Hayes-Roth 1981 for a discussion).

Section 2 discusses the need for implementating a natural language interface as part of AMTE/APU's research programme. Sections 3, 4 and 5 provide the background against which the current work should be viewed. Parsing and meaning are discussed in section 3; issues relating to "syntax" are presented in section 4; and section 5 looks at the "semantic" approach to natural language. The final section discusses the influence of existing parsers and parsing techniques on the design of AMTE/APU's natural language interface, and indicates the contribution of a natural language interface to the current work programme.

2. A NATURAL LANGUAGE SYSTEM AS PART OF AMTE/APU's RESEARCH

AMTE/APU are currently investigating several techniques of Knowledge Representation (see Sheppard 1981), which will be used to represent users' knowledge of a particular domain. The domain, HUNKS, is a simulated naval command and control task in which two simulated naval forces attempt to "seek and destroy" each other. Rowley (1982) provides a detailed account of the HUNKS domain.

Essential to any system representing knowledge is a point of communication between the user, acting on the particular domain, and the technique that is building the "user model":-

"The computer model of the user's domain knowledge is maintained as a dynamic data structure by an implemented Knowledge Representation System (KRS) - the modeller. The user model is the computer's "understanding" of the user (knowledge, intentions and constraints) at all times and without it the computer cannot provide intelligent support". (Sheppard op. cit.)

The natural language system, essentially, provides an "interface" between the user and the representation technique.

Although Sheppard (op. cit.) expands upon the systems currently being implemented, a brief summary at this point will indicate the importance of a natural language interface to one implementation in the work programme of AMTE/APU. The aim of this research is to build a representation of an individual's knowledge of HUNKS whilst that person is interacting with the domain. The KR technique requires the user to answer certain questions, in natural language, about his interactions with HUNKS. Natural language systems can be configured in a number of ways (see section 3) but most systems incorporate the implementation of a PARSER of some kind. The task of the parser in such a system is to analyse the natural language input and translate it into a form compatible with the internal representation. Thus if the input is something to be remembered by the system, it must be translated into a compatible form and stored, alternatively if the input is a question, the parser must analyse it into a form that can be used to guide an interrogation of the data-base for the appropriate answer.

In addition to a parser, the natural language system will access a LEXICON which consists of a list of all words "known" by the system. If, during input analysis, an unknown word is encountered, the natural language system must be able to return it to the user for definition and then add it to the lexicon. A more detailed exposition of the requirements of the natural language interface to be implemented by AMTE/APU will be given in the final section of this report.

3. NATURAL LANGUAGE SYSTEMS: PARSING AND MEANING

This Section is intended to provide the background against which the current research should be viewed. It addresses the issues that arise when attempting to define what is meant by understanding natural language. Winograd (1972) gives the following definition:-

"When a person sees or hears a sentence, he makes full use of his knowledge and intelligence to understand it. This includes not only grammar, but his knowledge about words, the context of the sentence, and most important, his understanding of the subject matter. To model this language understanding process in a computer, we need a program which combines grammar, semantics and reasoning in an intimate way, concentrating on their interaction".

Winograd's comments serve to indicate the enormity of the task. Such a "complete" natural language system has not yet been implemented.

Although each of the components mentioned by Winograd plays an important role in natural language understanding, early work in the area tended to concentrate on particular aspects, the consequence being a division between the "syntactic theories" and the "semantic theories" of natural language understanding.

The remainder of this Section will cover the most pertinent aspects of previous natural language understanding research, beginning with a discussion of what is actually meant by "parsing".

3.1 What Is Parsing?

It is difficult to define parsing without recourse to a notion of "structure", in the sense that there is a mapping between the item being "parsed" and a structure of some kind. However, it must be a particular structure, relating in some way to what are believed to be the essential characteristics of the item. For example, one could make a list of the number of letters in each word of a sentence (eg "the large red book" = "3 5 3 4") and call the resulting list "a structure", but the utility of such information is dubious! The "parse tree" has been the structure most common to the field of computational linguistics, an example of which is shown in Figure 3. A structure such as this is referred to as a "surface structure" because it is derived from the syntactic aspects of the input, that is, the word classes (noun, verb etc) rather than the word meanings.

The concept of parsing is also intimately connected with the act of "recognition", the determination of whether or not an item conforms to a particular specification. For example, is it an acceptable sentence according to the specified criteria? However, it is a mistake to view parsing and recognition as synonymous. A recogniser can only return a true/fail type response to an item, whereas a parser is expected to produce an output that differs from the input in a particular way. The output from the parser should be a structure that is somehow inherent in the item but is not apparent in its surface appearance. Thus the structure, as determined by the parser, is specific to the item being parsed in that the input item and the output structure are essentially two different representations of the same object.

Any definition of parsing must take account of the "purpose" of the exercise. Parsing is always done for some purpose, and the kind of structure required will depend upon that purpose. The same item may be represented by different structures depending on the purpose.

To be parsed, the item must conform to a particular specification. Therefore a set of rules is required to specify the criteria for accepting or rejecting an item. These rules are the GRAMMAR used by the parser and so the acceptability of any input is always relative to a particular grammar. The following definition of parsing offered by De Roeck (1981) further illustrates that parsing is a process:

"Parsing is the procedure by which a structure gets mapped onto (assigned to) a string. The thing that executes the instructions in the procedure is a parser. In our case it will be a computer program".

To summarise, a parser acts upon an input (in this case a string of English words) applies certain rules, and produces an output that can be used for a particular purpose. It should be noted that there is no universal agreement over the definition of parsing, but the definition given above underlies most of the research referenced in this report including that of AMTE/APU.

3.2 What Is Meaning?

In recent years, the issue of "meaning" has moved away from the realms of philosophy, and has been recognised as a practical problem:-

"If a machine is to understand questions and commands and take appropriate actions in response to them, then it needs some well-specified criteria for what these questions and commands mean." (Woods 1976)

Several factors influence the understanding of natural language including grammar, semantics, and reasoning. There has been a great deal of controversy about the relative contribution of syntax and semantics to the understanding of a sentence, although it is generally agreed that both play essential roles.

Thorne, Bratley and Dewer (1968) developed a parser, based only on syntax analysis, that could parse Lewis Carroll's "Jabberwocky", despite the fact that most of the words are nonsensical. However, a parser such as this is unable to check that the resulting parse is the correct one; this would require further analysis. A parser based on syntax analysis alone would have difficulty with ambiguous sentences such as:

"John hit the girl with the bat"

If the parser had access to syntactic information alone it would be unable to determine exactly who was in possession of the bat.

Clearly, syntactic analysis of sentence structure is important for understanding, but the need for additional analysis is easily demonstrated. This analysis is referred to as "semantic" analysis and once again there is no general consensus on what constitutes the field of semantics.

Researchers in this field tend to regard meaning and semantics as synonymous. Whereas a syntactic approach is concerned with the superficial manner of expression, a semantic approach to natural language understanding places emphasis on the underlying meanings or ideas expressed by the language. Semantic analysis of a sentence reveals three interdependent levels of meaning, word level, word-group level and sentence level.

3.2.1 Word level meaning

A requirement of any semantic analysis of language is the ability to define the meaning of the individual words in that language. Natural language understanding systems already implemented tend to store word definitions in some form of dictionary or lexicon. However, a single dictionary definition rarely captures the "full" meaning of a word or concept, consequently few researchers actually seek "absolute" definitions of words.

The definition of a word, as used by a particular natural language system, is the result of certain considerations:-

a. The actual parsing technique being used will influence the way a word is defined, depending on which aspects of a word the parser operates upon. For example, the word "John" could be defined as "pronoun", "agent" or "a particular male human", depending on the type of parser.

b. The words being defined must relate to some domain and the technique representing knowledge of that domain may have some bearing on the word definitions. Thus the more detailed the knowledge representation, the more detailed the word definitions are likely to be.

c. Finally, the previously discussed notion of "purpose" can be shown to be of equal importance to the definition of the individual words used by a particular system. Researchers are not usually interested in finding the absolute definition of a word and a word may have a number of meanings, with only one being relevant to a particular context. Since the purpose of most research in this field is to build a natural language system that "understands" inputs related to a particular domain, words need only be defined in terms of that domain. For example, DeJong (1979) gives three meanings for the word "fire": "to bake pottery in a kiln", "to terminate employment" and " to shoot". His purpose was to write a parser for newspaper articles, that could predict the meaning of words from their context. Thus the word "fire", for DeJong's purpose, would be assigned all of the above meanings.

It is interesting to note that word-level meaning naturally incorporates some notion of syntax. When word meanings include definitions such as "a type of action" or "a particular object", sentence analysis resembles the syntactic parsing of verbs and nouns. This serves to emphasise that there is no precise distinction between syntax and semantics.

3.2.2 Word-group level meaning

The second level of meaning, word-group level, refers to the meaning of a group of words within a particular syntactic structure. This level of meaning is, in fact, the province of the syntax analyst, showing that meaning is not necessarily synonymous with semantics. Analysis at this level looks at the ways in which syntactic structures convey meaning and at the role each word plays within that structure.

Sentences can be divided into various different structures, with different levels of complexity. The top-level structure is the clause, which determines the purpose of the utterance, ie whether it is a question, command or statement. The clause can be further divided into "noun", "verb", "preposition" and "adjective" groups. Each group has its own function in conveying meaning. Noun groups describe objects, verb groups convey information about time and mode of an event or action, prepositional groups describe simple relationships and adjectival groups work with the noun to describe objects and relationships. Each of these groups in turn can be examined at various levels of linguistic detail. Researchers in this field have to decide which level of detail is appropriate for their particular system. This decision is, in effect, the decision as to the type of parser to implement, as it is the parser that actually identifies the structures in question.

3.2.3 Sentence level meaning

At this level semantic theory focuses on how the meaning of a sentence depends on its "context". In semantic analysis, context can mean both the linguistic and the nonlinguistic setting of a sentence.

Linguistic setting refers to the context of the current discourse, that is, the relationship between the sentence currently being analysed and the preceding sentences. The following example illustrates this point:

"John went to the park. He took his kite".

If the second sentence was analysed in isolation, there would be no way of knowing that "he" and "his" referred to John. The meaning of anaphoric references, such as these, can only be discovered by recourse to linguistic context.

The nonlinguistic setting refers to any interaction between the current sentence and factual knowledge of the "real-world" or domain.
Consider the following sentence:

"His uncle ran the London marathon".

This sentence is syntactically unambiguous but without factual knowledge about "his uncle", the semantic meaning of the word "ran" is ambiguous - did his uncle organise the event or take part in it? The use of nonlinguistic context is one of the more difficult and controversial areas of natural language understanding research and as yet few existing systems can claim success in the area.

This discussion of meaning has shown that both syntactic and semantic analyses make significant contributions to the understanding of natural language. A theory of semantics cannot be applied to a sentence that is syntactically nonsensical and syntactic ambiguity can only be resolved by recourse to semantics. The following Sections 3 and 4 will review both the syntactic and the semantic approaches to natural language understanding with particular reference to the parsing techniques that represent each approach.

4. SYNTAX

4.1 What Is Syntax?

The previous Section focussed on how syntactic structures are able to convey meaning. This Section will explore the constituents of syntax, that is, the organization of strings of abstract symbols into a language. However, syntax can never be studied without the involvement of some notion of meaning:

"The structure of a sentence can be viewed as a series of syntactic choices made in generating it. The speaker encodes meaning by choosing to build the sentence with certain syntactic features, chosen from a limited set". (Winograd op.cit.)

Thus, although this Section aims to look only at syntax, the important role of meaning in the structure of language must be acknowledged.

Linguists have been studying the syntactic construction of language for a considerable number of years and have formulated rules which describe in detail how most sentences are constructed. Language can be seen as having a number of grammatical levels, which together comprise syntax. The following is a summary of these levels.

4.1.1 <u>The Word</u>

The basic unit of language is the individual word. However, a single word can have a variety of features which depend on its usage. For example, the word "walk" can be singular (walk), plural (walks), past-tense (walked), and a participle (walking). In addition, every word in the language belongs to one or more word classes, such as noun, verb, adjective, determiner, etc. The class of a word indicates how that word is likely to be used. For example, to describe something (adjective/adverb), the name of something (noun), or a type of action (verb).

4.1.2 <u>The Group</u>

The next syntactic level is the word group level, as previously described in Section 3.2.2. Certain groups, such as the "noun group" and the "verb group", are common to all theories of syntax, but there is no general consensus over the other word groups. For example, some theorists regard "prepositional groups" and "adjectival groups" as groups in their own right, whereas others treat them as constituents of noun groups. Each group has a number of word classes associated with it, certain classes being essential to the group, others being optional. An example will help to illustrate this point:

The following are all noun groups:

"The cat"
"The big cat"
"The big black cat"
"The two big black cats"
"Two big cats"

The noun "cat" is the only word common to all of the examples, showing that the only essential component of the noun group is the noun itself. Noun groups can have a determiner (the), and several adjectives (big/black/two), all of which are optional. The same is true of all groups at this level.

4.1.3 <u>The Clause</u>

Finally, the top syntactic level is the clause. The clause is treated as the highest grammatical level, as the sentence itself is more a unit of discourse than a syntactic structure.

A sentence can be either a single clause or a series of clauses joined with conjunctive words, such as "and". For example, "the boy went shopping" is a sentence with a single clause, whereas "the boy went shopping and bought a book" is an example of two clauses joined with a conjunction.

The clause is the most complex structure of the language, used to express relationships and events in a variety of different ways. For example, it can be a Question ("Who threw the ball?"), a Declarative ("He threw the ball"), or Imperative ("Throw the ball"); it can be Passive ("The ball was thrown by John") or Active ("John threw the ball"). Figure 1 shows only some of the features a clause can have, but for the current purpose, is an adequate illustration of the clause level of syntax.

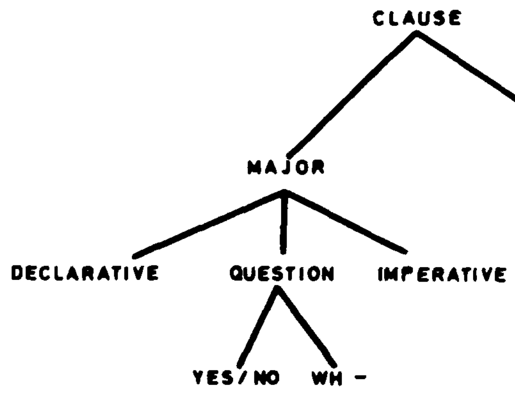


FIG. 1 THE CLAUSE LEVEL OF SYNTAX

This Section summarised the various grammatical levels that comprise syntax; in fact few sentences have such a simple three-levelled structure. Word groups often contain other word groups, and clauses can be part of other clauses. Thus an adequate theory or analysis of syntax is necessarily complex.

4.2 Grammar And Structure In Syntax Parsing

In Section 3.1, "grammar" and "structure" were shown to be important to the definition of parsing. In this Section, these terms will be examined in greater detail as they are central to any syntactical analysis of language.

One method of describing all sentences of a language is to provide a finite set of criteria that constrain the form of legal sentences; this is the precise role of the GRAMMAR which can be defined as:

"a set of rules that describe the sentences (or "strings") of a language and the structures that underlie those strings."

However, the role of the grammar is not to "produce" sentences of a particular language, but to describe and define them. The "recogniser" part of a parsing system performs a number of operations on an input, according to a given set of instructions. The rules of the grammar can be seen as the set of instructions used by the recogniser in deciding whether or not a particular sentence is part of the language.

Before further investigating the different types of grammar, it is necessary to note some of the conventional abbreviations and terms used in this area of work:

Terminal Symbols:

A grammar has a finite set of terminal symbols, which are essentially the words of the language the grammar describes. "Vt" represents the finite set of terminal symbols.

Non-terminal Symbols:

There is also a finite set of non-terminal symbols in a grammar, represented by "Vn". These correspond to the word-groups. For example, NG, VG, and PREPG are the non-terminal symbols for noun group, verb group and prepositional group, respectively.

Starting Symbol (S):

S is the symbol used to denote the beginning of a string.

Rewrite Rules:

Rewrite rules are similar to "production rules", and they have the form,

$$A \dashrightarrow B$$
(A rewrites as B)

- where both A and B are strings of terminal and/or non-terminal symbols.

These terms are commonly used in syntactic analysis of language and their use will be made clear in the following exposition of certain types of grammar.

4.3 Types of Grammar

In the field of syntactic parsing, most work has been based on the few grammar types to be covered in this Section. The grammars most frequently used in syntax parsing are the PHRASE STRUCTURE GRAMMARS, of which there are two types, CONTEXT-FREE and CONTEXT-SENSITIVE. Context, in this case, refers to the linguistic context WITHIN the input, not to the larger context of the domain.

4.3.1 Context-free grammars

A great deal of work on natural language parsing has involved the use of context-free grammars (CFG). CFG's are based on a set of rewrite rules of the kind mentioned above. The parser proceeds by taking a string of symbols and applying a rule to it which, essentially, rewrites part of that input string. The following is an example of a very simple CFG:

```
Vn= S, NG, VG, N, DET, V
    (non-terminals)
Vt= cat, mouse, the, eats,
    (terminals)
S--->  NG VG
NG-->  DET N
VG-->  V
VG-->  V NG
V--->  eats
N--->  cat
N--->  mouse
DET->  the
```

Fig 2a. A Context-Free Grammar.

This CFG describes the entire language, which consists only of the following sentences:

The cat eats.
The mouse eats.
The cat eats the mouse.
The mouse eats the cat.
The cat eats the cat.
The mouse eats the mouse.

Fig 2b. Sentences described by the CFG.

In addition to describing these sentences, the CFG also describes the structure underlying them. For example, "The cat eats the mouse" can be represented in the form of a "parse tree" (see figure 3).

A sentence has been parsed when all the structures that can be assigned to it, according to the grammar rules, are known. The structure such as the parse tree below, can be deduced by watching the steps taken during the application of the grammar rules.

4.3.2 Context-sensitive grammars

Context-sensitive grammars (CSG) are very similar to CFG's. They are also based on a finite set of rewrite rules that operate on the input string. The basic difference is that at least one (usually more) of its rules refers to symbols that have to occur

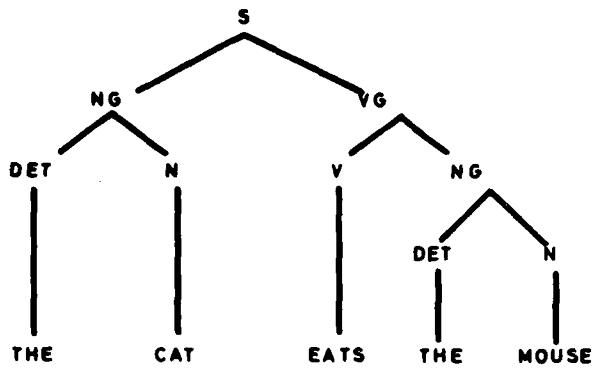


FIG. 3 STRUCTURE IN THE FORM OF A PARSE TREE

in a given order in the string before the rewritten rule becomes applicable. For example, the rule,

V---> terrify/NG

means that V can only be rewritten as "terrify" if terrify is followed by a noun group, which in this example, can be any noun group acceptable to the grammar. Thus, a context-sensitive grammar says very specific things about the syntax of the language. In the above example, because it only applies under certain conditions, the grammar rule acknowledges that "terrify" is a "transitive verb" and should be followed by a NG.

Context-sensitive grammar is very sensitive to certain features of an input such as grammatical context of particular words and certain word orders. For this reason CSG's will have a higher failure rate than CFG's when parsing inputs, because the grammar rules are more precise. CFG's are more flexible and consequently they are used more frequently in parsing systems. However, they are more prone to accepting inputs that are grammatically incorrect, in the spoken language sense. This is usually accepted in exchange for greater flexibility and application.

4.3.3 Transformational Generative Grammars

Transformational generative grammars (TGG) were first developed by Chomsky (1957) who believed that the phrase structure grammars failed to capture the essential properties of natural language. He argued that the most appropriate syntactic structure for an input may not be a tree directly corresponding to the surface words, but a "deep structure tree", which may require considerable re-arrangement of the surface structure. This deep structure is derived by applying a set of rules called "transformations" to the surface structure. Chomsky's intention was to use the deep structure tree to show all the significant information in a standardised format, whilst removing all superficial differences between sentences. For example, the deep structure for the sentence "The boy kicked the ball" is shown in Figure 4a, and the deep structure for "The ball was kicked by the boy" is shown in Figure 4b.

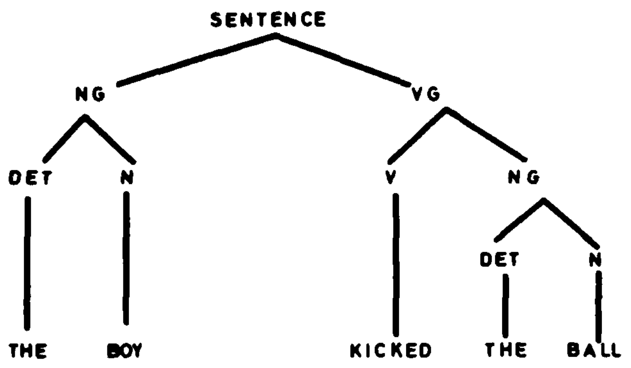


FIG. 4a DEEP STRUCTURE FOR "THE BOY KICKED THE BALL"

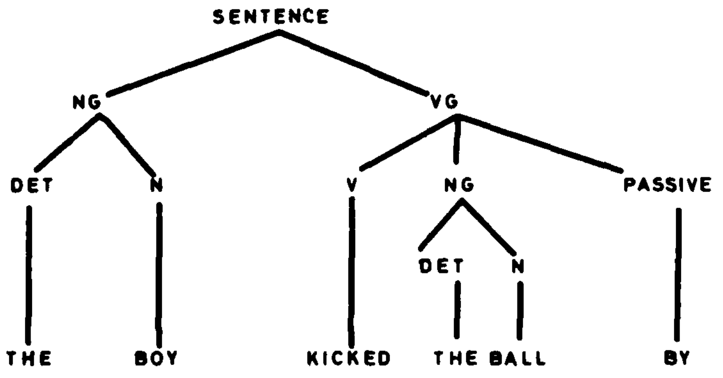


FIG. 4b DEEP STRUCTURE FOR "THE BALL WAS KICKED BY THE BOY"

Parsing, as previously defined, produces the underlying structure of the input string. TGG's are based on the notion of GENERATION which is essentially the reverse of parsing. Transformational rules are applied to the structure, in this case parse trees of the kind shown in Figure 3, until the tree is decomposed or "flattened" into a string, as defined by the grammar. Using the TGG for parsing is based on the idea that most inputs can be parsed in a number of ways, ie. several different structures can be built from the same input string. Typically, a context-free grammar is used to produce all possible parse trees which then serve as input to the transformational component:

> "...the trees produced from the string are meant to contain THE tree that could be produced from the transformational component when generating which, when flattened would give rise to this particular string." (King 1981)

To summarise, the "correct" parse/parse tree is identified by finding out which parse tree contains the grammatical categories needed to generate a particular sentence.

To achieve any degree of success, TGG based parsers would need a context-free grammar that could deal adequately with all possible sentences. Even if it was possible to implement such a grammar, the resulting TGG parser would be computationally time consuming, because it would inevitably produce large numbers of spurious trees, with no mechanism for rejecting the unlikely candidates. It is for this reason that TGG's have not been as successful as the phrase structure grammars.

4.4 Strategies for Grammar Parsing

There are basically two different ways in which a parser can proceed to assign a structure to an input and recognise the input as belonging to the language described by the grammar. It can either begin with the Start Symbol (S) and work down until it encounters a Terminal Symbol (an individual word), or it can start with the sentence and try to reduce it to S. The following context-free grammar will be used to illustrate both strategies:

(1) S---→	NG VG
(2) NG--→	DET N
(3) VG--→	V NG
(4) V---→	kicked
(5) N---→	boy
(6) DET-→	the
(7) N---→	ball
(8) VG--→	V

Fig 5. The rule set, to be applied to the sentence: "the boy kicked the ball".

4.4.1 Top-down Parsing

A top-down parser begins with S and expands it until a terminal symbol is reached. When this occurs, the parser moves to the next non-terminal symbol and continues until it reaches the end of the string. Thus in our example, rule (1) expands S as shown in Figure 6a.

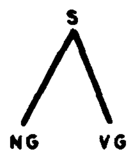


FIG. 6a TOP-DOWN EXPANSION OF S

The parser then checks to see if any of the new nodes are terminal symbols and as they are not, it expands the left-most node first by applying rule (2), shown in figure 6b.

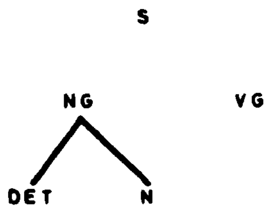


FIG. 6b TOP-DOWN EXPANSION OF NG

When both nodes are reduced to terminal nodes, the parser moves to VG, the next non-terminal symbol. To reach the correct parse, the rules should be applied in the order: 1, 2, 6, 5, 3, 4, 2, 6, 7, which gives rise to the parse tree in Figure 6c.

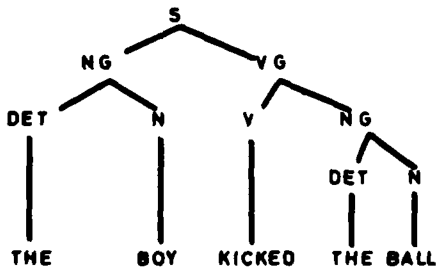


FIG. 6c TOP-DOWN PARSE TREE

However, there are two different rules that could be applied to expand N, namely, rules (5) and (7). If the wrong rule was applied, the terminal symbols in the parse tree would not correspond to the input string. The parser would discover the error when a check was made, either during the parse, or on completion of a string. If an error is detected, the parser has to "backtrack", that is, return to the point of error and try the other rules that could have been applied at that stage.

An alternative solution, known as "parallel parsing" involves building more one structure for each candidate rule. Although parallel parsing means that the input is only parsed once, with the correct parse being selected at the end, it tends to require a lot of computer memory and is, therefore, rarely implemented.

When, at any stage of the parsing procedure, more than one option for the next step is available, the procedure is said to be NON-DETERMINISTIC. Although most existing parsers are of this type, Marcus (1979) has implemented a parser that attempts to "simulate determinism". His parser avoids backtracking and parallel parsing by following a particular path to the end of the string and "flagging" any points at which ambiguity may arise. However, this technique does not remove the problem of rule choice, it merely deals with it in a different way.

4.4.2 Bottom-up Parsing

A bottom-up parser starts with the input string itself and tries to reduce it to the Start Symbol (S). This is achieved by replacing words with their category, then strings of categories with further categories, until S is reached. Thus the grammar of the previous example would have reversed rewrite rules, as follows:-

(1)	NG VG--->	S
(2)	DET N--->	NG
(3)	V NG---->	VG
(4)	Kicked--->	V
(5)	boy------>	N
(6)	the----->	DET
(7)	ball---->	N
(8)	V------->	VG

Fig 7. The Rewrite Rules For Bottom-up Parsing

A bottom-up parser using this set of rewrite rules, would start with the following sentence:

"the boy kicked the ball"

Reduction into categories can then proceed from either left or right, but this example will demonstrate that the same input parsed in different directions may yield quite different structures. Parsing from right to left, the first rule applied is (7), which rewrites "ball" as N. Rule (6) then rewrites "the" as DET, which allows rule (2) to rewrite the categories DET and N as a "higher level" category, NG. This proceeds until S is reached, and the resulting structure is the inverse of the parse tree arrived at by the top-down technique.

The same sentence parsed from left to right, however, leads to failure. Application of rules (6), (5), and (2) respectively rewrite "the boy" as NG, but then rule (4) changes "kicked" to V, rule (8) changes V to VG, and rule (1) arrives at S before reaching the end of the string, as in Figure 8.

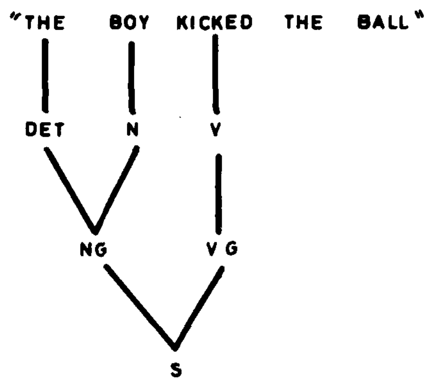


FIG. 8 BOTTOM-UP PARSE TREE

Although the parser has arrived at S, supposedly denoting the end of the input, there are unparsed words remaining - these would be revealed when the parse is checked (by a mechanism that looks to see if all words have been parsed). This indicates that techniques such as backtracking are necessary in bottom-up parsing as well as in top-down parsing, for recovery from erroneous parsings.

Top-down and bottom-up parsers are usually equivalent, because when they apply the same set of rules they will asign the same structures to the same sentences. The example used here was atypical, selected to demonstrate problems that can arise. The following Section will examine certain syntactic parsers in which such parsing techniques have been implemented.

4.5 Syntax-Based Parsers

This Section will concentrate on two syntactic parsers that have implemented the techniques previously described.

4.5.1 The Augmented Transition Network

The Augmented Transition Network (ATN) is probably the best known of the syntax parsing techniques, and has formed the basis for many natural language parsing systems. Woods (1970) is generally credited as the founder of ATNs, but his original 1970 paper acknowledges earlier work by Conway (1963) and Thorne, Bratley, and Dewar (1968).

The ATN is a way of representing the grammar of a language, designed to capture the regularities of natural language. The set of rewrite rules can act not only as a definition of a language, but also as a recognizer which accepts only the strings belonging to that language. Woods uses the term "automata" to refer to the use of the rules as recognizers, and suggests that the more complex the language, the more involved the recognition procedure is likely to be. The simplest automata are known as the FINITE STATE AUTOMATA, characterized by a set of STATES and ARCS, which show the possible orderings of constituents in a sentence and the various options the parser will have at any stage. The process is described by Ritchie (1981):

"The parsing program scans the network as it works through the sentence, making choices and carrying out actions as specified by the networks. The network grammar can be thought of as a highly specialised flow-diagram." (see Figure 9)

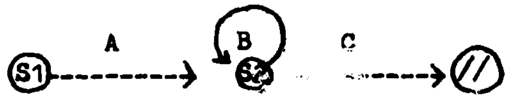


Fig 9. A Simple Transition Net.

S1, S2 and // are states, S1 representing the initial or start state. The arcs are transitions or "actions", and the labels on the arcs (A, B and C) give the conditions necessary in the input for a particular action to take place. The arc B indicates that more than one transition can be made at this point. For example, a NG can have multiple adjectives. Finally, nodes marked with diagonal bars are called FINAL STATES, indicating the completion of all transitions. The activity of the automaton is described succinctly by Johnson (1981):

"Computation begins in a designated state called an initial state (here S); the first symbol in the input sequence is examined to see whether there is a possible transition which matches that symbol; if there is, the machine follows the transition to the next state and "consumes" the input symbol - ie. shifts its attention to the next symbol in the sequence. This process continues until the machine "blocks" - ie. until no more transitions are possible. If the machine blocks in a final state and all the input has been consumed, we say the string has been ACCEPTED (or RECOGNIZED)".

The ATN is an extended version of a context-free recognition device called a RECURSIVE TRANSITION NETWORK (RTN). The main development of the RTN over its precursor, networks of automata, was the addition of a capacity for recursion. Such an extension is necessary because the finite state automata are not equipped to manage even the simplest form of recursion. Figure 10a shows a set of finite state automata corresponding to the rewrite rules in Figure 10b:

$$\begin{tikzcd}
\text{(a)} & \boxed{S1} \arrow[r, dashed, "NG"] & \boxed{S2} \arrow[r, dashed, "VG"] & \emptyset \\
\text{(b)} & \boxed{NG} \arrow[r, dashed, "DET"] & \boxed{S2} \arrow[r, dashed, "N"] & \emptyset \\
\text{(c)} & \boxed{VG} \arrow[r, dashed] & \boxed{S2} \arrow[r, dashed] & \emptyset
\end{tikzcd}$$

Fig 10a. Finite State Automata

(ai) S $\dashrightarrow$ NG VG
(bi) NG $\dashrightarrow$ DET N
(ci) VG $\dashrightarrow$ V NG

Fig 10b. The Corresponding Rewrite-rules.

To traverse (a) in Figre 10a, VG must pass control to (c), which then passes control to (b) before returning it to (a). The finite state automata have no mechanism for "remembering" where control came from or for returning control on completion of an embedded computation. Thus the RTN was developed to overcome these difficulties. Recursion in the RTN is enabled by adding a memory mechanism called a PUSHDOWN STACK which provides the means by which control can be transferred between different parts of the network.

The versatility of the RTN is increased or AUGMENTED by the addition of two important mechanisms:-

a. The ATN is equipped with a number of storage locations, usually referred to as REGISTERS, which are used to hold certain data during the parsing process. These registers are said to "augment" a network because their presence allows even the most syntactically complex of sentences to be parsed. For example, embedded phrases or clauses, common to English sentences, would be impossible to parse without recourse to resisters. Consider the sentence:

"I told her that you thought you knew John"

- not difficult for a human understander, but the parser would need to suspend the constituent it is currently working on, parse the inner phrases and resume where it left off. Thus the resisters provide a facility for storing partially completed structures whilst embedded phrases are parsed. Registers can also be used to postpone decisions about a particular symbol until more of the input has been seen. This is particularly useful when a word belongs to more than one category, eg "fire" can be a verb - "fire the gun!" or a noun - "the fire is burning". Registers can assist in assigning the correct category by holding the word in memory until more of the input has been parsed.

b. Augmentation is also achieved by the addition of a "structure-building function". The RTN is basically a strategy for context-free recognition and is not a very useful parsing device as it lacks a means of producing a structured output. The RTN is, therefore, further augmented by the addition of a structure-building function which constructs a parse tree structure simultaneously with the process of recognition. Thus one of the actions associated with the arcs of an ATN, activates the structure-building function. The output, the actual parse, is the value returned when the final state is reached.

The above description of the ATN outlines only the basic design features, the actual implementation is influenced by the individual programmer and the host computer language.

Users of the ATN make no claims about its validity as a psychological model of human language processing, and it has been used mainly in experimental AI work. The arguments in support of the ATN are varied, as are the criticisms; the following is a representative summary:

a. Backtracking:

A common criticism of ATNs is associated with the problem of backtracking, as described in Section 4.4.1 ATNs normally proceed in a "depth-first" manner, following only one path. The ATN, therefore, is highly dependent on backtracking when recovering from an incorrect path, and this can be very time consuming.

b. Clarity of Notation:

Woods (1970) emphasizes what he refers to as the "perspicuity" of ATN notation and is critical of other grammars. He says of Tranformational Grammars:

"It is not possible in this (TG) model to look at a single rule and be immediately aware of its consequences for the types of construction that are possible. The effect of a given rule is intimately bound up with interrelation to other rules.... The Augmented Transition Network provides the power of a Tranformational Grammar but maintains much of the perspicuousness of the context-free grammar model".

Relative ease of writing and the ability to capture many related pieces of linguistic knowledge simultaneously are two advantages of ATN notation, but given a complex ATN complete with resisters and structure building functions and the notation may not be as clear as Woods' claim suggests.

c. Control:

The ATN, like procedural models in general, allows the details of parsing strategy to be tightly controlled. The price paid in return is the danger that as the system increases in size, modifications become difficult to control and debugging procedures tend to create more problems than they remove.

4.5.2 Predictive Analysis Parsing

The Predictive Analysis model of parsing natural language was orginated by Kimball (1973), in an attempt to take account of the limitations of the human perceiver. He proposed a number of related principles for the parsing of English syntax based on the notion of sentence "acceptability" and in particular, on what renders a sentence difficult to understand. Whaley (1979) has implemented some of Kimball's principles in his syntax parser PA/O (Predictive Analysis with no "look-ahead") which also shows an interesting application of some of the techniques presented in the earlier discussion on types of grammar (Section 4.3).

The parser is essentially top-down, staring with the Start Symbol (S) and working down the non-terminal nodes to the terminal symbols. As words are processed, branches are formed from new or existing non-terminal nodes down to the new terminal symbols. In Kimball's model, non-terminal nodes are "flagged" as either OPEN or CLOSED as a function of certain linguistic factors. Whaley (1979) explains:

> "consider the word "borrowed" in a sentence that begins "The girl borrowed..." the tree just prior to the perception of the word "borrowed" consists of the S-node to which an NG (noun group) node is attached. Below the NG-node are the terminal symbols DET ("the") and N ("girl"). NG is the lowest non-terminal node; and it is OPEN prior to the occurrence of the verb "borrowed" since it is possible that an additional phrase or clause could follow. However, since a verb, for linguistic reasons, cannot be attached to an NG-node, NG is now CLOSED and a new node, VG, is created and attached to S."

Thus, the tree representation, as given in Figure 11, begins to resemble those in figure 6.

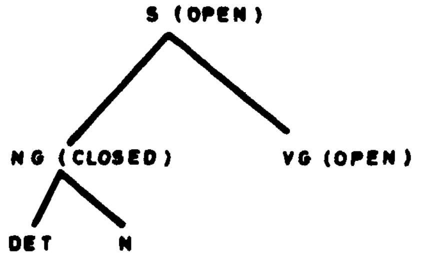


FIG. 11 TOP-DOWN PARSE TREE OUTPUT FROM PA/O

The parsing process quickly runs into difficulties if too many nodes are allowed to remain open at one time. The parser therefore, operates on the principle that a new node is created when the next word parsed has a different grammatical function to that of the existing open node. The node then remains open until a terminal symbol, that is not an immediate constituent of that node, is parsed. Such a terminal symbol simultaneously closes the preceeding node and opens a new one, which is immediately attached to the "lowest open non-terminal node". The following example will illustrate the parsing process. Consider the following context-free grammar:

$$\begin{array}{ll} S \dashrightarrow & NG\ VG \\ NG \dashrightarrow & DET\ N \\ VG \dashrightarrow & V\ ADVERB \\ VG \dashrightarrow & ADVERB\ V \end{array}$$

The S-node remains open until the parse is complete, thus the first node created is the NG-node which is attached to S. The DET and N are attached to the NG-node as they are parsed, the NG-node being the lowest open non-terminal. The NG-node remains open until either a V or an ADVERB is encountered. At this point the NG-node is closed and a VG-node created which becomes attached to S, as S is now the lowest open node. This procedure continues until S is the only open node and all words of the input string have been parsed. Output from PA/O is in the form of "surface structure diagrams" - a form of parse tree, and at any time the user can ask for "status information" which shows which nodes are open or closed and how much of the input has been parsed.

There is always the danger of closing a node prematurely, as grammar rules cannot anticipate all grammatical possibilities. For example:

"She liked the boy she met at the party very much."

To assign "very much" to the correct node, the parser would need a very sophisticated "look-ahead" facility, ie. the ability to suspend decisions about the current non-terminal symbol until more of the input has been seen. Whaley feels that a two word look-ahead would cope adequately with most English sentences, and is planning to add such a facility to his existing PA/O algorithm.

The PA/O parser has several other features, in addition to its parsing capacity, that enhance its flexibility. The user, via a series of commands, is allowed to:

"Create and display its lexicon, in part or total.

Modify the lexicon. Items may be added, deleted, or re-ordered within the terminal symbol categories that structure the lexicon.

Enter a sentence and modify it. One can study the consequences of major or minor modifications of a sentence on the parsing process.

Batch-process large numbers of sentences. Whole files of sentences, in paragraphs or list form may be submitted to the parser." (Whaley op.cit.)

The performance of PA/O is impressive. The parser is able to proceed even when a number of words in the sentence are not found in the lexicon. Dummy symbols ("???") are assigned to the nearest preceding open node, and successful parsings have been produced when up to five words have been missing. PA/O can parse sentences with a wide variety of syntactic structures, and with varying degrees of complexity.

In conclusion, Whaley's PA/O parser is one of the more recent and successful syntax parsers. It demonstrates a number of features that are currently under consideration at AMTE/APU, and the extended version (PA/2) with its look-ahead facility, will further increase its versatility.

This Section has presented an overview of syntax including techniques of syntax parsing and certain implementations of these techniques. The next Section will take a similar look at the field of semantics and semantic parsing.

5. SEMANTICS

5.1 What are Semantics?

During the 1960s, syntactic analysis of natural language was the central issue in computational linguistics, with surprisingly little discussion of semantics. This trend was reversed during the early 1970s with the emergence of a number of "non-syntactic" approaches to language understanding. It is perhaps, more appropriate to use "semantic" to mean "non-syntactic", as very little agreement exists over the definition of semantics.

The Section on meaning (Section 3.2) pointed out that meaning and semantics are usually regarded as synonymous by researchers in this field, and in many definitions, semantics are described as "the underlying MEANING of words and sentences". Ramsey and Atwood (1975) in a review of the field of man-computer interaction, suggest that human communication itself is characterized by ungrammatical utterances and syntactic ambiguity. They believe understanding is achieved by recourse to semantics. They say of natural language systems:-

"Satisfactory natural language processing requires that the computer has considerable semantic information about the application domain and about prior conversation in order to

disambiguate, and ultimately to comprehend the users
statements"

Whilst they suggest the kind of information an Understanding System should access, they fail to specify the form this information should take, ie, what is meant by "semantic information" in terms of practical implementation? Unfortunately, this omission is not uncommon in this area of research!

There have been four main approaches to the practical issues underlying semantics, SEMANTIC PRIMITIVES, SEMANTIC NETWORKS, SLOT GRAMMARS and PROCEDURAL SEMANTICS, all of which will be covered in the following Sections.

Wilkes (1977) distinguishes between two types of semantic parsing systems. The "superficial" parsing systems, such as the pattern matching programs (PARRY, Colby 1975; ELIZA, Weizenbaum 1966) have very little semantic content, but rely on a particular methodology to give the appearance of understanding. "Deep" parsing systems, however, aim to parse directly from the input to a semantic representation of some kind, without undergoing any conventional syntactic analysis. The approaches to be discussed in the following Sections all fall into the "deep" parsing category.

5.2 Strategies for Semantic Parsing

This Section presents four strategies that have been developed for the semantic analysis of natural language, semantic primitives, semantic networks, slot grammars and procedural semantics. However, there is a large overlap between these approaches. For example semantic networks are based on a set of semantic primitives and slot grammars were developed from the semantic network research. This Section will review each approach separately, as different problems arise in each.

5.2.1 Semantic Primitives

The "decomposition" of language into primitives was first suggested in the early 1960s by Katz and Fodor (1963) - one of the first theories of semantic structure. Meaning, they believed, could be expressed in terms of some standard set of features or relations, and all meanings could be represented by some configuration of these basic elements. At first, "binary" semantic features were used to classify simple objects. For example, an object can be either "+ animate" (animate) or "- animate" (inanimate). It soon became clear that a simple "conglomeration" of these binary features was an inadequate representation of relational information, and consequently sets of primitive relations were introduced. Fodor (1970) suggested that "John Killed Mary" could be represented (in the notation of the AI programming language LISP) as:-

```
CAUSE(JOHN, BECOME(NOT(ALIVE(MARY))))
```

This approach has been strongly criticised by those who believe this to be merely "a re-arrangement of the English words with some (meaningless) notion thrown in" (Ritchie 1981). Relational primitives can appear to represent simple relationships such as:-

"John likes Mary" = LIKES(JOHN, MARY)

- where a certain relation (LIKE) exists between two entities (John and Mary). This form of representation certainly fails in more complex situations. For example, "John left the shop without buying anything" could be represented as:-

BEFORE(LEAVE(JOHN, THE(SHOP)), NOT(BUY(JOHN, ANYTHING)))

Not only is it difficult to describe "anything" and "leave" as entities, but also "before" and "buy" are doubtful as relations! Ritchie (1981) warns against the belief that:

"...you have captured the "meanings" because you have scribbled a plausible definition; calling your relations "primitive" does not make them magically meaningful"

The work of Schank (1972, 1975) on "Conceptual Dependency" is perhaps the best known in the field of semantic analysis of language. His ideas evolved from an interest in human understanding in general, and have been applied specifically to language understanding. Schank's contention is that human understanding is a process by which new information is assimilated in terms of the old information already present in memory (see Schank 1975 for a discussion on his theory of memory). His work encompasses a theory of "primitive ACTs" which suggests that all language and speech can be broken down into elemental units of meaning:

"It is useful to restrict severely the concepts of actions such that actions are separated from the states that result from those actions....Simply stated, the theory of primitive ACTs states that within a well-defined meaning representation it is possible to use as few as eleven ACTs as building blocks which can combine with a larger number of states to represent the verbs and abstract nouns in a language," (Schank 1972)

In Conceptual Dependency theory there are three main components: (i) ACTs are things that animate objects do to physical objects, which usually result in a state change, (ii) STATES are the conditions present that allow an action to take place, eg, "Mary rode her bike" indicates several states including, possession-of-bike, ability-to-ride etc and (iii) the RELATIONAL LINKS between ACTs and states which are of two basic types; the relation between an actor and an action, and between an object and an attribute. As ACTs can only refer to the things animate objects do to physical objects, this formalism is rather restricted in terms of what constitutes an ACT. Within the limits of this representation, all actions can be derived from only eleven ACTs, these include:

"ATRANS: The transfer of an abstract relationship such as possessions, ownership or control.
INGEST: Animate object transfers something to his inside, ie. food, liquid, gas.
GRASP: The grasping of an object by an actor.
MTRANS: Transfer of mental information between animals or within an animal.
MBUILD: The construction by an animal of new information from old information, including imagination, decisions, realizations etc." (Schank 1975)

Schank represented natural language sentences in terms of objects, ACTs and states, using "Conceptual Dependency diagrams" as the method of representing the structure of the sentence. For example, the sentence "John liked the food" would be represented as in Figure 12.

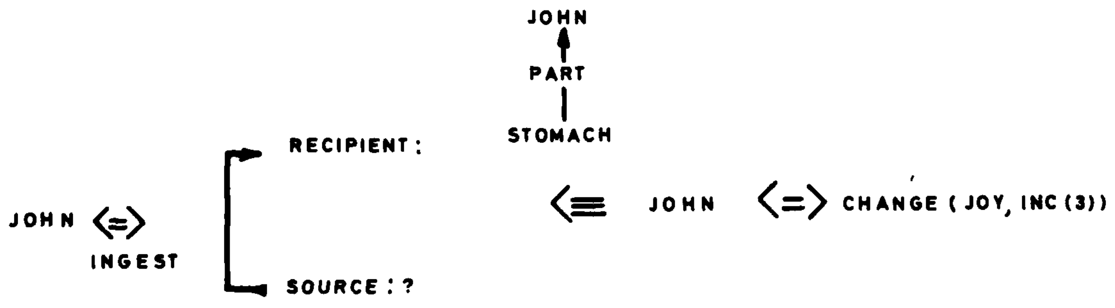


FIG. 12 CONCEPTUAL DEPENDENCY DIAGRAM FOR THE SENTENCE "JOHN LIKED THE FOOD"

In Schank's opinion, the advantage of this approach is that similarities in meaning can be captured by structures having similar or identical subparts, he feels that this allows inferences to be easily made. In Figure 12, the dependency diagram lends itself to the inference that the food is in John's stomach. However, Ritchie (1981) feels that until a set of rules is provided that carry out such matches and inferences, the dependency diagrams are no improvement on Fodor's "primitive formulae" cited previously.

The main criticism of any approach based only on "primitives" is that they do not accept larger units as relevant to understanding. Norman and Rumelhart (1975) believe that although the "primitive" approach lends itself to simple sentences, a more "global" approach is necessary for the understanding of more complex passages. They suggest that primitives should still represent meaning at the lowest level, but that larger units (corresponding to words, phrases or general rules) should also be incorporated into Language Understanding Systems.

Although Conceptual Dependency has not been implemented as part of a natural language System, both Semantic Network and Slot Grammar theorists recognise the influence of Schank on their research.

5.2.2 Semantic Networks

Semantic network theory originated from the Ph.D thesis of Quillian (1968) and has since become one of the most popular techniques for representing meaning. Quillian's original aim was to represent the semantics of English words in the form of a graphical "network" of relations between "entities". There is a similarity between the Semantic Primitive approach and Semantic Network theory, in that both represent interrlations by labelled graphs, but Semantic Network theories tend to incorporate certain claims as to the structure of human knowledge and memory. Unfortunately, these claims vary considerably between theories rendering the field of semantic networks generally confusing.

Brachman (1979) has examined the history of semantic networks in an attempt to clarify what is actually represented by such networks and whether the claims made about them are substantiated. In his general conclusions he says the following:

> "Through the ten-year history, at least five different interpretations of nodes and links have crept together to create confusing languages with limited expressive power. In the last two years, efforts have been mounted to crack that expressive deadlock; these efforts have concentrated on the logical status of network primitives, and have begun to take a hard look at the foundations of network representations. At the same time, the field has begun to see higher-level structures imposed on nodes and links. These structures appear to be useful and significant, but no comprehensive effort has been made to understand exactly what their status is."

The following account of the semantic network approach to language understanding is intended to summarize the main issues involved in such a heterogenous area of research.

Quillian (op.cit) first proposed a representation composed of "nodes" interconnected by various kinds of "associative links". Not unlike the organization of a dictionary, the nodes are "word concepts" and links from a node point to other word concepts, which together constitute a definition of the original word concept. The structure, therefore, is an interwoven network of nodes and links. Quillian uses the term "plane" to refer to the structure that holds the definition of a word concept, thus if a particular word can have three senses, the network would incorporate three "planes". The example originally given by Quillian presents three meanings of the word PLANT, each enclosed in its own plane; links within a plane form the structure of the definition; links out from a plane indicate other planes in which all referenced words are defined. Figure 13 is a simplified illustration of Quillian's planes:-

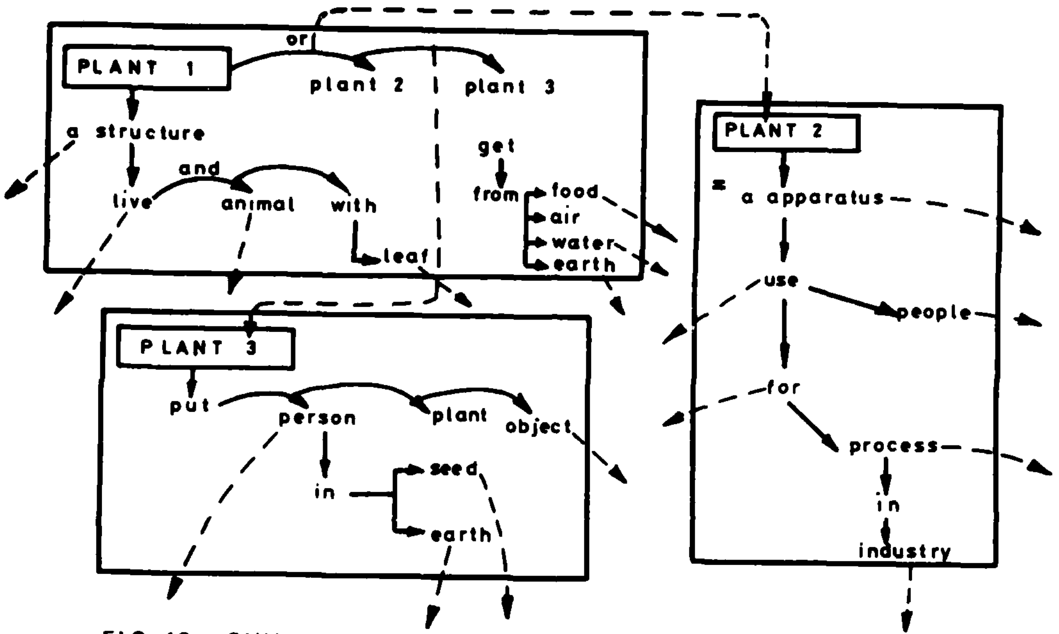


FIG.13 QUILLIAN'S PLANES, A SIMPLE SEMANTIC NETWORK

Qullian proposed that the full meaning of any word concept could be found by tracing down through the network from the original (defining) plane.

This work first introduced a notion that later became known as "inheritance", and which is now accepted as central to all network theories. In the original formulation, a particular type of link - the "subclass" link - was used to enable certain inferences to be made about the properties of an object. This link indicated a subclass relationship between a specific concept and a more general concept, thus properties that were deemed true of a class were assumed true of all its subclasses, and properties of a subclass could be inferred by reference to the general class. "Inheritance", therefore, refers to the passing of values for particular properties from the general class to the specific case. For example, the general class "dog" may have the properties mammal, quadruped and canine, which are inherited by the specific case "terrier". The advantage of such a mechanism is that large amounts of information can be located in just one position in the network. However, disadvantages arise when a specific case inherits only some of the properties; that is, should the inference process consider all possible inheritable properties for every node it encounters, and if not, how can selective inference be implemented? This is an issue still in contention.

Carbonell (1970) applied the ideas of Quillian by using the network as a data structure in his computer-aided instruction program, SCHOLAR. The network held the system's factual knowledge (the geography of South America), and students asked questions about the data-base. This work made two significant contributions to the original network theory. Firstly, distinctions were made between "conceptual units" (eg. city, latitude) and "example units" - particular examples of concepts (eg. Argentina). Secondly, this work extended the notion of inheritance by introducing the process of "instantiation" - the actual construction of an individual description from its "generic" description, ie the process by which an individual becomes recognized as "an instance of" a particular class.

Another interesting approach to semantic networks was that of Fillmore (1968) who focused his networks on the verb. This "case structure" approach has the action (verb) as the node and the entities (usually referred to as the "agent", the "participants" and the "object") as links. Previous nets have used noun-orientated nodes and verb-orientated links. For example, a network representation of the sentence "John gave Mary the book" could have "gave" as the node, "John" as an agent-link, "Mary" as a participant-link and "book" as an object-link.

This description of the case structure approach is an extremely simplified one, intended to give a general view of the area, many different types of links have been implemented in various systems (Rumelhart and Norman 1973; Rieger 1976; Anderson and Bower 1974). This approach is particularly significant because it can be viewed as the precursor of the more recent Slot Grammar approach.

In conclusion, semantic networks attempt to represent the semantics of English words via the "concepts" that correspond to words and sentences. The main problems associated with this approach can be demonstrated simply by pointing out that no two pieces of research have identified the same links. Each network is based on some theory of which primitive elements constitute a concept, and no agreement exists over the primitives underlying language (English or otherwise). As Brachman (1979) says:

> "Since the semantics of any given language is dependent on the interpretation of the primitive elements and a set of rules for combining them into nonprimitive elements, the "well-definedness" of a network language rests heavily on the set of node and link types it provides."

He continues to suggest that although a straightforward comparison of primitives used in different research is impossible, it is possible to identify a small number of distinctive types of node and link. He has identified five "levels" of semantic network, Implementational, Logical, Epistemological, Conceptual and Linguistic. Each level has a particular set of link types, and a network implementation can combine any of the levels. For a detailed account of the suggested levels see Brachman op.cit. It is Brachman's belief that if a general understanding of each type of primitive was reached, any formalism of semantic network could be clearly and completely specified, even if it combines elements of more than one level. To achieve a useful semantic network representation the emphasis should be on using a small, well-defined set of primitive node and link types, whose operations are consistent at a particular level.

5.2.3 Slot Grammars

An interesting, more recent development in semantic analysis of language, is the slot grammar technique, which has its roots in semantic network theory. Interest in this technique originated from the work of Minsky (1975) who developed a theory of FRAMES, for the representation of knowledge. Frame theory was developed as a general representation technique, the field of Language Understanding being only one of a number of areas to which this technique has been applied.

Prior to discussing the original ideas of Minsky, it is interesting to note that various research departments in this field have introduced their own terms for similar techniques. Minsky's term FRAMES continues to be used by those at the Massachusetts Institute of Technology, the Stanford University AI community refers to SCRIPTS and more recently, the term "UNIT PACKAGE" has been applied to this technique (Stefik 1977). The term "slot grammars" is used specifically when applying the technique to the domain of Language Understanding, and will be used to refer to the general Frame/Script theory and methodology.

The original formulation of Frame theory by Minsky was an attempt to move away from the prevailing "knowledge primitives" approach. He believed that humans do not constantly refer to "first principles" (primitives) in order to "know" something. He postulated that people have vast chunks of stereotype knowledge about previously encountered situations, which allows them to make assumptions about what is "normal" in such situations:

> "When one encounters a new situation (or makes a substantial change in one's view of a problem), one selects from memory a structure called a FRAME. This is a remembered framework to be adapted to fit reality by changing details as necessary. A FRAME is a data-structure for representing a stereotyped situation like being in a certain kind of living room or going to a child's birthday party. Attached to each frame are several kinds of information. Some of this information is about how to use the frame. Some is about what one can expect to happen next. Some is about what to do if these

expectations are not confirmed." (Minsky, 1975)

Frames, like semantic networks, are networks of nodes and relations. SLOTS are essential to the frame data-structure and hold the data that link the nodes. The difference between this technique and the semantic networks lies in the type of data involved. As Minsky has pointed out, slots may be filled with various types of data, not only factual data (as would be found in a semantic network) but also PROCEDURAL data (eg. steps to be taken if one's expectations fail). It is this integration of factual, or "declarative" data with the procedures necessary for manipulating that data that has generated such interest in the slot grammar technique.

Although a great deal has been written about frame representation, actual implementations vary considerably. Basically, frames hold descriptions of objects or actions, and each slot contains a particular value for a particular property of that object or action. There may be restrictions on the type of value a slot may hold. For example, a frame for the action "walk" may have a slot entitled "done-by" that requires its value to be the name of an animate object. Slots may also be filled with "default" values, which prevail until replaced with new values, as when new situations are encountered that modify an existing frame for that situation. Finally, slots can be filled with "pointers" to procedures that are triggered when that slot is filled with a particular value. For example, Winograd (1975) describes a frame-based representation of a calender domain; if asked "what day is June 28th this year?" a "what-day" slot is filled with the value "June 28" which triggers the appropriate calculation.

A frame network is usually organized hierarchically, and involves the previously mentioned notions of inheritance and instantiation. This "generalization" hierarchy forms a structure of "IS-A" links connecting particular concepts to other concepts of which they are "specializations". For example, in a frame representation of a simple natural language system, "dog" would have an IS-A link to "noun"; conversely, "noun" could have a link called "HAS-INSTANCES" to "dog". Thus inheritance implies that any property true of a concept in the hierarchy is also true of any concepts linked below it, unless contradictions are explicitly stated.

The use of slot grammars for natural language understanding became popular in the late 1970s because of the similarity between case grammars and frame representations. The term "slot grammar" refers to the representation of actions in terms of frames and slots. The name of an action (ie, the verb in a sentence) becomes the frame name, and the slots correspond to the objects associated with the verb. Figure 14 shows a case grammar represented as a slot grammar.

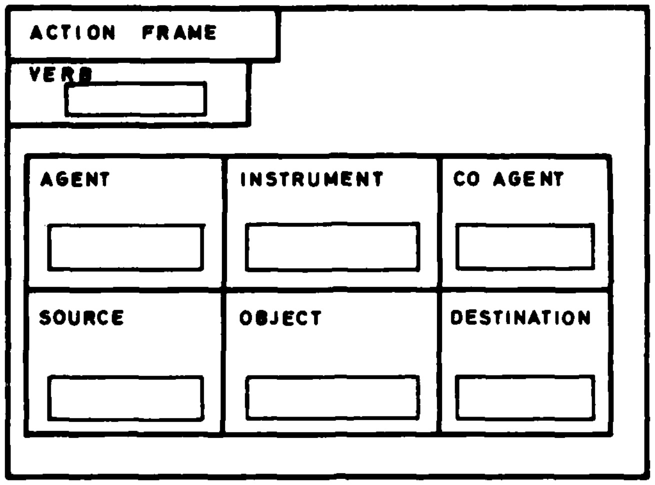


FIG. 14 SLOT GRAMMAR REPRESENTATION OF A GENERAL CASE GRAMMAR

The more specific verbs can be represented by having certain slots filled by default values (eg, the verbs "speak", "talk" "whisper" etc would have the "agent" - or "actor" slot as it is often called - filled with the default "human"). To fully understand a sentence using this form of representation, action-frames would have links to "state-change" frames, thus enabling the consequence of actions to be represented. Figure 15 shows how Winston (1977) represents "Moving the pyramid onto a red block made Robbie happy".

Slot grammars and frames have formed the basis of many investigations into natural language understanding. Charniak (1972) used the term "world knowledge" to refer to the stereotype situations used in the understanding of children's stories; Hayes (1977) has superimposed a frame structure over a semantic network for use with a natural language system for finding the correct senses of ambiguous words in particular contexts. Recently a number of representation languages specifically for the implementation of frames and slot grammars have emerged. These languages include Frame Representation Language (FRL) Roberts and Goldstein (1977), Knowledge Representation Language (KRL) Bobrow and Winograd (1976), and Network Language (NETL) Fahlman (1975). Developments such as these have considerably increased the interest in this representation technique.

Wilkes (1975) work on SEMANTIC PREFERENCE is closely related to the slot grammar technique. Wilkes allies his research to frame representation because both rely on structuring knowledge into units larger than the primitives:

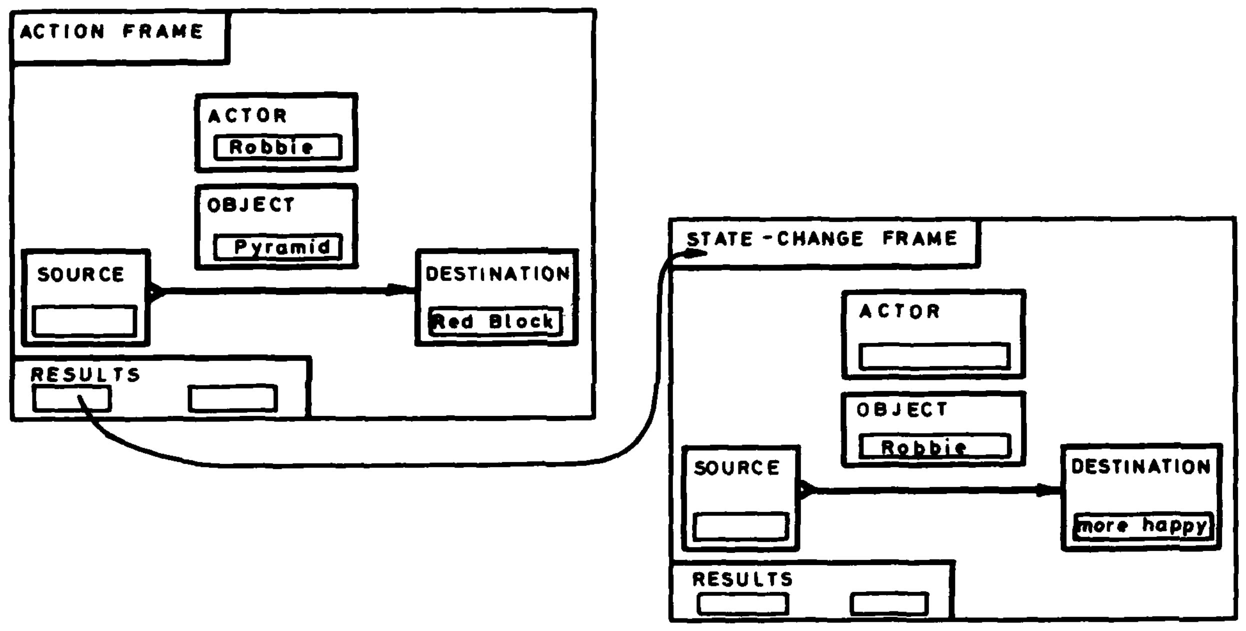


FIG. 15 CAUSE AND EFFECT FRAME NETWORK, FOR "MOVING THE RED PYRAMID ONTO A RED BLOCK MADE ROBBIE HAPPY." (WINSTON 1977)

"There is a fairly well-defined set of basic messages that people always want to convey whenever they write and speak, and in order to analyse and express the content of discourse, it is these simple messages that we need to locate." (Wiles, 1975)

These basic messages he refers to as GISTS, and his semantic analyser has the task of identifying the "gists" of input sentences. Wilkes has constructed a "semantic dictionary" using a scheme of eighty semantic primitive elements. For example, the definition of "drink" would appear as:

"an action preferably done by animate things, to liquids, causing the liquid to be in the animate thing, through a particular part of the animate thing"

Such a description is not dissimilar to the semantic primitives of Schank. The semantic primitive elements combine to form TEMPLATES, which comprise agent, actor and object triples. Each word has a FORMULA, one for each different sense of the word. The templates, therefore, seek an actor, action and object in every sentence, and as the words that fill these requirements are themselves individual formulae, the sentence analyser reveals an entire network of formulae. The sentence analyser proceeds by first segmenting the text in clauses, and then applying the templates to the various segments, seeking a "template match". It is Wilkes contention that any part of the input that cannot be matched to a template is not semantically meaningful. However, it could be argued that it is not unmeaningful only in terms of this particular representation.

Wilks' representation of sentence meaning resembles slot grammar technique in that the templates seek actor-action-object formulae, as frames have actor-action-object slots; and just as a frame system has restrictions on the contents of certain slots, Wilks' formulae can be seen as active entities, dictating how the sentence analysis should proceed. The final process in the analysis ties all templates together to form one "semantic block", which consitutes the meaning of the sentence.

It is difficult to draw any final conclusions about the slot grammar technique as implementations are still in their infancy. In general, it provides a methodical approach to sentence analysis and the results are promising, especially when the application domain remains reasonably simple. There are still many problems to be solved, such as representing the meaning of analogies, metaphors and creative writing in general. These problems, however, apply to all areas of natural language understanding and not just to this approach.

5.2.4 <u>Procedural Semantics</u>

For a number of years, the field of Artificial Intelligence has been the 'battleground' for a very controversial issue - the

DECLARATIVE versus the PROCEDURAL approaches to AI. The declarative approach holds that knowledge should be represented as a large number of "facts" in a data base, with a set of general procedures for manipulating that data base. The proceduralists propose that certain kinds of knowledge (such as reasoning, deduction and heuristic search) can be represented more efficiently in terms of executable procedures. This approach is referred to as "the procedural embedding of knowledge".

Within the area of natural language understanding the controversy continues and the procedural arguments are echoed in the "procedural semantic" approach to meaning. This approach asserts that "knowing-what" is equivalent to "knowing-how", that is, something is "known" only when there is a procedure for either describing that something (if it is an object) or for doing that something (if it is an action).

The procedural semantic approach is best illustrated by the work of Winograd (1972). His domain, SHRDLU, is a simulated world of blocks, pyramids and boxes, that can be manipulated by a simulated robot arm. Based on this domain, the natural language system can answer questions, execute commands and accept information in an interactive English dialogue.

Winograd uses a theory of procedural semantics to represent the knowledge in his system:

> "Knowledge in the system is represented in the form of procedures, rather than tables of rules or lists of patterns. By developing special procedural representations for syntax, semantics and inference, we gain flexibility and power. Since each piece of knowledge can be a procedure, it can call directly on any other piece of knowledge in the system." (Winograd, 1972)

The procedural approach focuses on "how" facts are used, rather than on the facts themselves. Its power lies in the fact that all procedures have to be "integrated", such that each procedure "knows" enough about the other procedures to enable their appropriate use as Winograd states, each piece of knowledge can call any other piece of knowledge. Unfortunately, the source of power is also the source of the main problem, namely, how to add new information without having to modify all existing procedures. One solution, as suggested in the work of Rumelhart and Norman (1973), is to leave procedures as "open" as possible to the addition of new information. For example their system has a procedure for representing the meaning of the word "the", which searches the context of an input for an example of the item in question. Thus, the meaning of "the red box" is a procedure which searches for a recently mentioned red box. The procedure is "open" in that it can, without modification, be applied to any referenced object.

SHRDLU had an overwhelming impact on natural language understanding research, and (superficially) it remains one of the most impressive programs in the field of Artificial Intelligence. As Ritchie (1981) says:

"SHRDLU was a virtuoso feat of programming, containing some useful and provocative ideas, but it did not provide the solution to any general problems. Hence AI was struggling with a distorted view of what was possible, and overlooked the fact that we still do not have an adequate theory of language."

In a review of the procedural approach, Ritchie (op.cit.) concludes that although it is a useful way of looking at certain types of words, some refinement of the theory is necessary. He suggests that in order to view language understanding as a form of processing, it is important to form a clear idea of what meaning is, something few researchers have achieved. Woods (1975) is more critical, he feels that it is not sufficient to equate the meaning of a sentence with the procedure executed in response to it:

"in general, the procedural semantics is a paradigm or a framework for developing and expressing theories of meaning, rather than being a theory of meaning itself." (Woods, 1975)

Davies and Isard (1972) further propose that if language understanding is to be viewed entirely as a process, it is essential to take account of the effect sentences create in the hearer/rader. They believe the process does not stop when the sentence is perceived by the hearer, and account should be taken of what the hearer does with a meaning under particular circumstances.

Clearly, a number of problems arise when formulating a purely procedural approach to natural language understanding. The increased interest in slot grammar/frame theory indicates that researchers are considering a synthesis of both procedural and declarative approaches as a possible alternative. Simon (1969) believes that "understanding", the ultimate aim of AI, is basically the relationship between the representation technique, one or more bodies of knowledge and the set of tasks the system has to perform. He suggests that most domains combine a variety of different types of knowledge, such as, simple facts, general knowledge about reasoning, and procedures of deduction for question-answering. He concludes:

"Knowledge without appropriate procedure for its use is dumb, and procedure without suitable knowledge is blind".

This Section completes the review of the existing research in the field of natural language understanding. Sections 3, 4, and 5 have presented an account of the various approaches to the understanding of language, including the current achievements and failures. In conclusion, the aims of AI work are threefold: i) to enrich the body of knowledge available to the system (the facts), ii) to enrich the procedures that use that knowledge in the performance of an ever increasing range of tasks, iii) to enrich the quality of communication between the user and the system - including natural language communication.

6. AN EXPERIMENTAL IMPLEMENTATION OF A NATURAL LANGUAGE INTERFACE

The research presented in this state-of-the-art review of natural language systems forms the background against which the current work should be viewed. This Section will consider the requirements for implementing a natural language interface as part of APU's research on Intelligent Support Systems. To do this, it is necessary to examine exactly what is required of the interface, ie, to look at the type of interactions involved, as these will have a considerable influence on the design. This Section will also discuss those features of existing parsers that will influence the design and subsequent implementation of an interface.

6.1 Design Considerations

The natural language interface is an essential component of the "HUNKS UNDERSTANDER" currently being implemented at AMTE/APU. Sheppard (1981) provides an account of this work which aims to implement "on-line modelling of knowledge and skills used in command and control" (Sheppard op.cit). Towards this aim, the UNDERSTANDER will combine several techniques including "Personalised Task Representation" (PTR), a technique for representing an individual's task knowledge (Gregory 1979, 1981); the concept of "primitives" as units of representation (see Section 5.2.1); and a data structure of FRAMES, as discussed in Section 5.2.3.

6.1.1 Personalised Task Representation

A detailed account of PTR's role as part of the HUNKS UNDERSTANDER is presented by Gregory (1981). PTR has its origins in Task Analysis, and was formulated as a method of directly interrogating experts about the way they perform certain operations. More recently, however, developments in the theory of PTR have revealed its potential as a language for the representation of knowledge. Gregory (op.cit.) summarizes the central aim of PTR:

"When we encounter an object" (concept, action, procedure, activity, task etc) about which we want to learn, we may try to discover what it is by asking what it is for, how it is brought about and when its use is appropriate. The main point is that the answers to these questions do not exist independently of each other, but are systematically related to the object in question. The function of the PTR structure is to formalise these apparently natural relationships."

PTR aims to elicit users' personal descriptions of their actions by pursuing a particular line of questioning, namely, HOW an action was performed, WHY that particular action was selected (the purpose) and WHEN that action is appropriate according to the elicitee, taking into account the environmental factors that the elicitee believes to be important conditions for the action.

6.1.2 Primitives

The Schankian notion of primitives is founded on the principle that human understanding is a process by which new information is assimilated in terms of the old information already present in memory. A similar principle underlies the use of primitives as basic units in HUNKS, a domain which is (intentionally) uncomplicated, with a restricted number of possible actions permitted to the user. Rowley (1982) provides a complete specification of the HUNKS command simulation.

Briefly, there are only three different commands users can give to the vessels under their control, these give rise to the "action" primitives "move", "ping" and "fire". In addition there are "intelligence" primitives such as "active detection" and "explosion", that are part of the simulation and occur as a result of users' interactions with the HUNKS environment. There is, therefore, a limited number of primitives associated with the actual "playing" of HUNKS, although there will also be a set of "parsing" primitives (see Section 6.2.1). Nevertheless, it is anticipated that users will be allowed to explain their actions using concepts of their own choosing, the requirement being that to be 'executable', these concepts should be defined ultimately in terms of the HUNKS primitives:

"By anchoring the subsequently elicited representations in task dependent primitives, the system's potential for machine execution of the representation can be realised." (Gregory op.cit)

Implementation of an executable representation gives rise to certain issues, notably the Procedural/Declarative controversy - the representation of knowledge as procedures or as facts (as discussed in 5.2.4). Executability of the knowledge in the representation is of major importance in the HUNKS UNDERSTANDER, as it provides the means of achieving "prescriptive" understanding of users' actions. In order to demonstrate its understanding, the representation system must be able to execute the same procedures as would users, under similar circumstances (ie. by executing a representation of the answers given by the user in response to the PTR questions).

It is essential to recognise the distinction between users' knowledge and the knowledge in the representation - they are not necessarily equivalent. The claim is that an action is only "known" by the system if there are appropriate procedures for executing that action (as in SHRDLU Winograd 1972): no such claim is being made about the USER'S knowledge. The distinction is important because this approach avoids many of the criticisms that have been made about the procedural approach as a theory of human memory (see Section 5.2.4).

6.1.3 Frames

The use of primitives has the advantage of allowing certain knowledge to be represented as executable procedures. The need for a procedural component also underlies the implementation of a frame-based data structure (the user model) as part of the HUNKS UNDERSTANDER. Recalling Section 5.2.3, frames (Minsky 1975) enable the integration of factual (declarative) data with the procedures necessary for manipulating that data. Gregory (1981) presents an account of the logical relations of PTR expressed in terms of frame methodology. Briefly, the slots in each frame correspond to the PTR questions and the frames are linked by the relations defined by PTR, namely, purpose, method and cause. The lexical component forms part of the declarative aspect of the frame network and will be discussed in the following section.

6.2 A Frame-Based Natural Language Interface

Section 6.1.1 on PTR discussed the type of interactions that will involve the interface, and the frame-based data structure being implemented was covered in Section 6.1.3. This Section considers the implementation of a natural language interface for the HUNKS UNDERSTANDER, in the light of these design considerations.

6.2.1 The Lexicon

Section 6.1.2 proposed the use of a limited set of primitives that define the HUNKS domain. These primitives could be referred to as the "playing" primitives, that is, they are the only ones necessary for playing HUNKS. However, when users are required to explain their actions (via the interface), the playing primitives alone become an insufficient language for communication. The aim of any natural language interface is to allow users as much freedom of description as possible, thus the more words "understood" by the system, the more natural the interaction will seem. Towards this end, the interface proposed will have a LEXICON as an integral part of the user model's frame-based data structure. The lexicon will be a set of "lexical frames" that are purely declarative, that is, they are simply a factual set of the words "known" by the system and are not executable, as are the "action frames". The data base, therefore, has both a "prescriptive" (procedural) element, comprising the action frames, and a "descriptive" (declarative) element, the lexical frames being only part of the descriptive component. In addition to the lexical frames, there will be a set of descriptive frames that simply "describe" the concepts introduced by the user. These will link certain descriptive information to the action frames, but are not themselves actionable. For example, the frame for the concept CONTACT could be linked to the action frames for "explosion" and "detection" by HAS-INSTANCES links (explosions and detections are examples of the concept contact) but the frame for "contact" is simply descriptive, not prescriptive.

Each concept (referred to as a "node") that is understood by the system will have a number of frames associated with it, including a lexical frame. A node can comprise a single word or a complete phrase, but every word has a lexical frame, including words such as "and", "the", etc which have a purely parsing function. When an input is being parsed, the parser will check that each word in that input has a lexical frame, and if not, that word is flagged as unknown. The parser will incorporate a mechanism for returning such words to the user for definition. This mechanism is the means by which users can introduce new concepts. New concepts (see section 6.1.2) to be executable, must be defined ultimately in terms of the HUNKS primitives, thus when the user defines new concepts, links are formed between the descriptive and the prescriptive aspects of the data base.

6.2.2 Slot Grammar Parsing

A slot grammar is currently being implemented as the technique of parsing (see Section 5.2.3). This approach is particularly appropriate for two reasons. Firstly, slot grammars are the language understanding systems' equivalent of frame methodology and secondly, slot grammar parsing focuses on the verb, which is particularly concordant with the action-orientated HUNKS domain.

Implementation details are not yet finalised, although a number of possibilities are being considered. As slot grammars seek an action, actor and object in each sentence, it seems reasonable to have a "sentence frame" with action, actor and object slots. Parsing, therefore, can be seen as the process by which these slots are filled, one frame for each sentence of the input.

To accomplish such a parsing process, the lexical frame for each word will have a "node-type" slot, indicating whether a word is an "action-node" (ie, a verb like MOVE, or FIRE), an "object-node" (ie, a noun like VESSEL or HOSTILE) or a "parse-node" (words such as AND, THE, AM etc). Users' inputs (provided they remain reasonably uncomplicated) would typically have one action, one actor and one object, so the basic parsing process is one of checking the lexical-frames for the node-type of each word, locating the action, actor and object and putting them into the appropriate slots in the sentence-frame.

The definition of unknown words by users is dependent upon how words are to be represented. To be actionable, the definitional process will ultimately create links between the new word and the HUNKS primitives. However, lexical definition amounts to creating a new frame for the word and identifying its node-type. It is anticipated that contextual information will aid lexical definition. For example if a "bottom-up" technique is employed (Section 4.4.2) the first step would be to find the node-type of each word. If an unknown word is encountered, its potential node-type can be inferred from the context of the remainder of the input. Thus, if an actor and an object are already present in the input, it can be inferred that the unknown is an action. In addition, there should be a mechanism for returning the input (or a paraphrase) to the user for confirmation of node-type.

The parsing of prepositional groups is a problem that has already been identified. Certain words such as "towards" and "behind" and phrases like "away from" and "close to", are directly relevant to the actions in HUNKS, but are not verbs (actions) or nouns (objects) and yet should have greater significance than the parse-nodes. There are two possible ways of dealing with prepositions such as these. Firstly, they could be assigned a completely distinct node-type, such as "action/prep-node". Secondly, they could be treated as part of the action-nodes. In the former case, each node would be a single word, with separate nodes for prepositions and verbs. In the latter case, complete phrases would become single nodes (eg "move-towards" and "fire-at"). Design issues such as these are currently being considered. Figure 16 is a diagrammatic summary of the ideas outlined in this Section.

6.3 Existing Language Parsers

The proposed natural language interface presented in the previous Sections is being designed and implemented from first principles at AMTE/APU. Although previous research has influenced the current work considerably, it was decided not to implement an existing parsing system for several reasons:

i. The computing facilities available at AMTE/APU restrict the implementation of certain existing parsers. For example, the Augmented Transition Networks that have become popular in the USA run on computer architecture not available to AMTE/APU.

ii. The principle underlying the use of a domain such as HUNKS was to create an uncomplicated task environment, where all variables are known, such that the domain will not interfere with the basic research. The same principle can be applied to the implementation of a natural language interface, that is, such a restricted domain does not need a particularly sophisticated interface, and a complicated parser could needlessly hinder the other research issues.

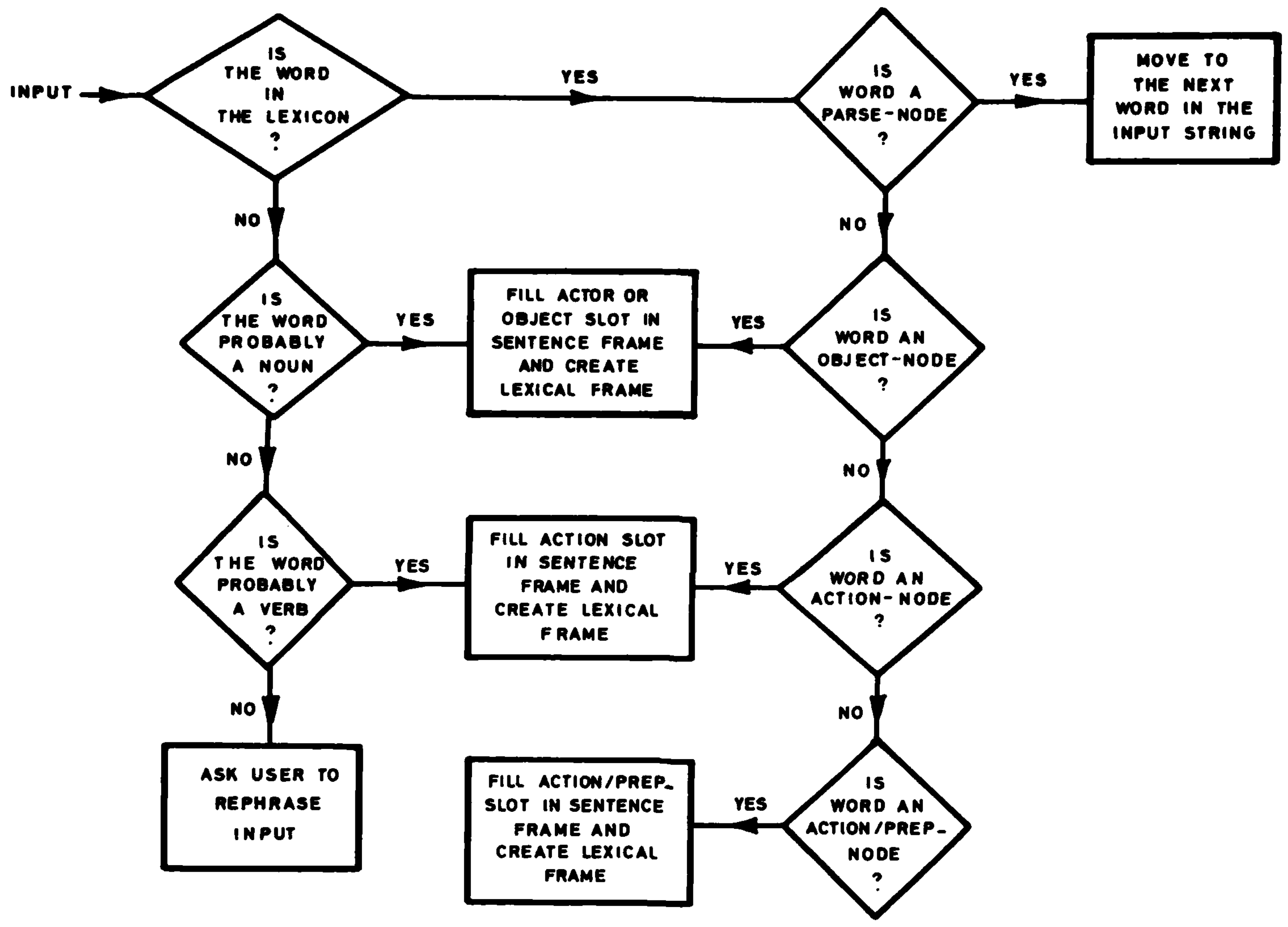

INPUT
IS THE WORD IN THE LEXICON ?
YES
NO
IS WORD A PARSE-NODE ?
YES
NO
MOVE TO THE NEXT WORD IN THE INPUT STRING
IS THE WORD PROBABLY A NOUN ?
YES
NO
FILL ACTOR OR OBJECT SLOT IN SENTENCE FRAME AND CREATE LEXICAL FRAME
YES
IS WORD AN OBJECT-NODE ?
NO
IS THE WORD PROBABLY A VERB ?
YES
NO
FILL ACTION SLOT IN SENTENCE FRAME AND CREATE LEXICAL FRAME
YES
IS WORD AN ACTION-NODE ?
NO
ASK USER TO REPHRASE INPUT
FILL ACTION/PREP. SLOT IN SENTENCE FRAME AND CREATE LEXICAL FRAME
YES
IS WORD AN ACTION/PREP. NODE ?

FIG. 16 LEXICAL DEFINITIONS OF UNKNOWN WORDS

iii. The majority of successful parsers already in existence are too domain-specific to be used at AMTE/APU. The SOPHIE system (Burton and Brown 1975) is a prime example, SOPHIE is a program for tutoring electronic troubleshooting and is similar to AMTE/APU's proposed system in that it combines procedural knowledge in the form of "procedural specialists" with factual knowledge in a semantic network. It uses a "semantic grammar" to analyse natural language inputs, which recognises entities with certain semantic properties. This is made feasible by incorporating much of the world knowledge of the domain into the grammar rules. Although the parser is very successful, it is extremely domain dependent, and extensive rewriting of the grammar would be necessary before it could be used with a different domain.

7. SUMMARY

This report provides a comprehensive review of the field of parsing natural language. The relative merits of syntactic and semantic parsing systems are discussed, and several problems associated with the term "meaning" in natural language research are identified. Various parsing techniques are covered in some detail, and several existing parsing systems are discussed.

The report also outlines the approach taken by AMTE/APU towards the implementation of a natural language interface for the HUNKS UNDERSTANDER system. The major influences on the design of an interface are identified, and finally reasons are given for deciding against the implementation of any of the parsing systems already in existence.

The work discussed in this report represents a very large area of research: the study of natural language understanding within the context of Artificial Intelligence. Consequently, there is a great deal of research that cannot be included in a report such as this, and the main aim is to cover (at least) the major milestones.

M. B. ROWLEY (Psychologist)

MBR/SGD

REFERENCES

ANDERSON, J. R. & BOWER, G. H. A propositional theory of recognition memory. Memory & Cognition, 2, (1974)

BOBROW, D. & WINOGRAD, T. An overview of KRL: A knowledge representation language. Cognitive Science (Cog Sci), 1, (1977)

BOBROW, D. & WINOGRAD, T. KRL: Another perspective, Cog Sci, 3, (1979)

BRACHMAN, R. R. On the epistomological status of Semantic Networks. In Findler, N (Ed) Associative Networks. New York Academic Press, (1979)

BROWN, D. G. & BURTON, R. R. Multiple representations of knowledge for tutorial reasoning. In Bobrow, D. & Collins, A. (Eds) Representation and Understanding. New York Academic Press, (1975)

CARBONELL, J. R. AI in CAI: An Artificial Intelligence approach to Computer-aided Instruction. Int J Man-Machine Studies (IJMMS), 11, (1970)

CHARNIAK, E. Towards a model of children's story comprehension. Mass. Int. Technology (MIT). AI Lab. Report, AD 755 232. (1972)

CHOMSKY, N. Syntactic Structures. Mouton Press, (1957)

COLBY, K. M. Computer simulation of a neurotic process. In Tomkins & Samuel (Eds) Computer Simulation of Personality: Frontier of Psychological Research. Wiley, (1963)

COLBY, K. M. Artificial Paranoia. New York Press (1975)

CONWAY, M. E. Design of a separable transition-diagram compiler. Communications of the Association for Computing Machinery, 6, (1963)

DAVIES, D. J. M. & ISARD, S. D. Utterances as programs. In Meltzer & Michie (Eds), Machine Intelligence, 7, (1972)

DeJONG, G. Prediction and substantiation: A new approach to Natural Language Processing. Cog Sci, 3, (1979)

De ROECK, A. An underview of parsing. Conference notes, Tutorial on Parsing Natural Language, Castagnola, Switzerland. (1981)

FAHLMAN, S. E. NETL: A system for representing real world knowledge. MIT Press, (1979)

FILLMORE, C. J. The case for case. In Bach & Harms (Eds), Universals of linguistic theory, Holt, Rinehart & Winston (1968)

FODOR, J. A. Three reasons for not deriving "kill" from "cause to die". Linguistic Inquiry, 1, (1970) GREGORY, R. Personalised Task Representation. Tech. Memo AMTE(E)TM 79103. AMTE, Teddington. (1979)

GREGORY, R. Personalised Task Represntation: Developments. Tech. Memo AMTE(E)TM 81104. AMTE, Teddington. (1981)

HAYES, P. J. On semantic nets, frames and associations. Int. Conf. On AI, (1977)

HAYES-ROTH, P. A. A tutorial on Expert Systems: Putting knowledge to work. In.Jnt.Conf. on AI, Vancouver, Canada (August 1981)

JOHNSON, R. Parsing with transition nets. Conference notes, Tutorial on Parsing Natural Language, Castagnola, Switzerland, (1981)

KATZ, J. J. & FODOR, J. A. The structure of a theory of Natural Language Systems. Language, 39, (1963)

KIMBALL, J. Seven principles of surface structure parsing in Natural Language. Cognition, 2, (1973)

KING, M. Transformational grammar parsing. Conference notes, Tutorial on Parsing Natural Language, Castagnola, Switzerland, (1981)

KUIPERS, B . J. A frame for frames: Representing knowledge for recognition. In Bobrow & Collins (Eds) Representation and Understanding, New York Press, (1975)

LEHNERT, W. & WILKES, Y. A critical perspective on KRL. Cog Sci, (1979)

LEVESQUE, H. & MYLOPOULOS, J. A procedural semantics for semantic networks. In Findler, N. (Ed) Associative Networks, New York Press, (1979)

MARCUS, M. P. A theory of syntactic recognition for Natural Language. (1979 thesis) MIT Press (1980)

MINSKY, M. A framework for representing knowledge. In Winston, P. (Ed) The Psychology of Computer Vision. McGraw-Hill, (1975)

NORMAN, D. A. & RUMELHART, D. E. Explorations in cognition. Freeman Press, (1975)

QUILLIAN, R. Semantic Memory. In Minsky M. (Ed). Semantic Information Processing. MIT Press (1968).

RAMSEY, H.R. & ATWOOD, M. E. Human factors in computer systems: A review of the literature. Science Applications Inc, USA, Tech. report, AD-A071 597 (1979)

RITCHIE, G. History of Natural Language Processing, issues and methods. Tutorial notes, Tutorial On Artificial Intelligence techniques. Milton Keynes, (1981) RIEGER, C. An organization of knowledge for problem solving and language comprehension. Artificial Intelligence, 7, (1976)

ROBERTS, B. B. & GOLDSTEIN, I. P. The FRL manual. MIT AI Lab, AD A052 310. (1977)

ROWLEY, M. B. HUNKS: Design and rationale, (AMTE(E) TM82101) AMTE Teddington, (in preparation 1982)

SCHANK, R. Conceptual dependency: A theory of Natural Language Systems. Cog Psychol, 3, (1972)

SCHANK, R. The primitive ACTs of conceptual dependency. Theoretical Issues in Natural Language processing, 1, (1975)

SHEPPARD, C. Applied Psychology Unit Man-Computer Studies Section Rationale and Work programme 1981/82. E1/P4.1/198/81 AMTE Teddington, (1981)

SIMON, H. The sciences of the artificial. MIT Press, (1969)

STEFIK, M. J. An examination of a frame structured representation system. Proceedings of Sixth Int.Jnt.Conf. on AI. (1979)

THORNE, J. P., BRATLEY, P. & DEWER, H. The syntactic analysis of English by machine. In Michie (Ed) Machine Intelligence, 3, Edinburgh Univ. Press, (1968)

WEIZENBAUM, J. ELIZA: A computer program for the study of Natural Language communication between man and machine. Communications of the Association for Computing Machinery, 9, (1966)

WHALEY, C. P. Predictive Analysis in sentence comprehension: A computer simulation model for surface structure parsing. IJMMS, 13, (1980)

WILKES, Y. Preference semantics. In Keenan (Ed) Formal semantics of Natural Language, Cambridge Univ. Press, (1975)

WILKES, Y. Good and bad arguments for semantic primitives. AI Dept. Univ. Edinburgh, (1977)

WILKES, Y. Making preferences more active. In Findler, N. (Ed) Associative Networks, New York Press, (1979)

WINOGRAD, T. Procedures as a representation for data in a computer program for understanding Natural Language. MIT AI Lab. AD-722 399. (1971)

WINOGRAD, T. Understanding Natural Language. Cog Psych. 3, (1972)

WINOGRAD, T. Five lectures on Artificial Intelligence. Stanford Univ. AI Lab, AD-A000 085, (1974)

WINOGRAD, T. Frame representations and the Declarative-Procedural controversy. In Bobrow & Collins (Eds) Representation and Understanding, New York Press, (1975) WINSTON, P. H. Artificial Intelligence. Addison-Wesley, (1977)

WOODS, W. A. Transition network grammars for Natural Language analysis. Communications of the Association for computing machinery, 13, (1970).

WOODS, W. A., KAPLAN, R. M. & NASH WEBBER, B. The Lunar sciences Natural Language information system. BBN rep. N72-28984 (1972)

WOODS, W. A. What's in a link: Foundations for Semantic Networks. In Bobrow & Collins (Eds) Representation and Understanding. New York Press, (1975)

WOODS, W. A. Speech understanding systems: Final report. BBN rep. 3438 (1976)

DATE
ILME